REMARKS

This paper is responsive to a non-final Office action dated September 23, 2004. Claims 1-26 were examined. Applicant has amended claim 1.

Interview Summary

Applicant appreciates the Interview on December 21, 2004, which included participants Steven R. Gilliam and Todd Ingberg. The *Aho* and *Peake* references were discussed. No agreement as to allowability was reached.

Rejections under 35 U.S.C. §101

The Office has rejected claims 1 – 10, 12 – 23, and 26 under 35 U.S.C. §101 as being directed to non-statutory subject matter. Applicant respectfully traverses the rejections.

Applicant respectfully submits that claims 1 and 26 are directed to statutory subject matter because a multipass parser (claim 1) and an apparatus (claim 26) are products, and a product is statutory subject matter. However, Applicant has amended claim 1 to include the term "product."

With regard to claims 13 and 23, these claims are directed to processes, which is statutory subject matter. The subject matter of independent claims 13 and 23 produce "a concrete, tangible, and useful result" (MPEP 2106, citing AT&T Corp. v. Excel Communications, Inc., 172 F3d 1352 at 1358 (Fed. Cir. 1999)). Both of the claims 13 and 23 recite limitations that disclose resulting parse states, as in claim 13, or resulting abstract syntax trees, as in claim 23. The parse states and the abstract syntax tress are concrete, tangible, and useful. Both the parse states and the abstract syntax trees include data that is generated and utilized in parsing, and are not mere abstract concepts or numbers being manipulated. Hence, the subject matter of the claims are directed to statutory subject matter.

Rejections under 35 U.S.C. §102 and §103

The Office has rejected claims 1 – 26 under 35 U.S.C. §102(b) as being anticipated by "Support for Modular Parsing in Software Reengineering" by Peake et al. (*Peake*). The Office

offers an alternative rejection of claims 1 – 26 under 35 U.S.C. §103 as being obvious in view of the combination of *Peake* and "Compilers Principles Tools and Techniques" by Aho et al. (*Aho*). Applicant respectfully traverses these rejections.

Peake fails to teach the subject matter of the claims, despite combination with *Aho*. *Peake* applies software engineering principles to the construction of a parser. *Peake* states that their "narrowed goal is to create parsers whose code can be reused as though they were modules" (p. 59). Peake then states the paper does "***not consider issues of parse tree construction***" (p. 59). Hence, Peake discloses modular parsing with specific reference to combinatory parsing, and even discloses parsers for specific grammars implemented as collections of functions (p. 59). The addition of *Aho* only provides disclosure of traditional top-down recursive descent parsing, whereas the claimed subject matter facilitates decomposing parsing into multiple simple passes. Therefore, the combination of *Peake* and *Aho* arguably suggests implementing a modularized parser that performs traditional top-down recursive descent parsing tree construction.

The combination of *Aho* and *Peake* neither disclose or suggest miniparsers that successively operate on abstract syntax trees or parse states resulting form a predecessor miniparser. The Office refers to page 61 of Peake as disclosing successive operation on abstract syntax trees. However, the relied upon section actually discloses results of two parsers being concatenated, which does not disclose or suggest successive operation on respective abstract syntax trees resulting from respective predecessor miniparsers.

In addition, although *Peake* discloses implementing modular parsers for specific grammars, it does not limit parsers to particular subsets of syntactic constructs. The new trees and particular subsets of syntactic constructs, lead to immutable subtrees which grant succeeding miniparsers freedom to reuse the subtrees resulting from predecessor minparsers. This paradigm facilitates easier debugging and a priori elimination of bugs. Referring to paragraph 43 of the specification:

> Since a miniparser does not alter its input, both its input and output are available for inspection when debugging a miniparser. And since parse trees are immutable, a class

of bugs resulting from mutating shared objects is eliminated.

Claim 1 recites "plural miniparsers each successively operable on a respective abstract syntax tree...that includes transformations of predecessor ones, if any, of the miniparsers", which is similarly recited in claims 13 and 23. The Office refers to sections of *Peake* that disclose plural parsers that concatenate their results and sections of *Aho* that disclose abstract syntax trees, but, as already stated, *Aho* merely discloses traditional tree construction. Nothing in either reference discloses or suggests parsers that generate new trees, each of which includes transformations of predecessor trees of the predecessor parsers. Claim 1 also recites "wherein respective ones of the miniparsers are limited to particular subsets of syntactic constructs to be parsed", which is also similarly recited in claims 13, 23, and 26. The Office refers to a section of *Peake* that discloses a facility of Common LISP Object System (CLOS) for generic functions, which are declared with signatures that stipulate the class of the parameters. The disclosure of a generic function and signatures that specify subclasses does not suggest or teach limiting miniparsers to particular subsets of syntactic constructs. Moreover, *Peake* specifically states that parsers are for specific grammars, not subsets of a grammar.

With regard to claims 2 and 3, the Office rejects the claim based on a section of *Peake* that discloses combining parsers to make new parsers. However, claim 2 claims respective abstract syntax trees being output by miniparsers. Claim 3 includes the recitation that "the respective input and output abstract syntax trees are separately encoded." The combination of parsers to make a new parser does not suggest or teach miniparsers outputting respective abstract syntax trees, or the respective abstract syntax trees being separately encoded.

With regard to claim 6, the Office relies on a section of *Peake* that discloses extending a grammar. However, the relied upon section does not disclose or suggest the limitation that "none of the miniparsers individually implements the grammar." As previously stated, *Peake* states that its parsers are for specific grammars.

With regard to claim 8, the Office relies upon a section of *Aho* that discloses lexical analysis and various sections of *Peake* that disclose token generation and combining parsers to

make a new parser. None of these sections, or any other section of *Aho* or *Peake*, disclose or suggest parsers with functionality as disclosed in claim 8.

For at least the reasons above, neither *Peake* nor *Aho*, standing alone or in combination, disclose or suggest any of Applicant's claims. In addition, the dependent claims are allowable at least because they depend from respective ones of the allowable independent claims.

Conclusion

In summary, claims 1-26 are in the case. All claims are believed to be allowable over the art of record, and a Notice of Allowance to that effect is respectfully solicited. Nonetheless, if any issues remain that could be more efficiently handled by telephone, the Examiner is requested to call the undersigned at the number listed below.

CERTIFICATE OF MAILING OR TRANSMISSION

I hereby certify that, on the date shown below, this correspondence is being

☐ deposited with the US Postal Service with sufficient postage as first class mail, in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

☒ facsimile transmitted to the US Patent and Trademark Office.

Steven R. Gilliam — Dec. 23, 2004 (Date)

EXPRESS MAIL LABEL: _____

Respectfully submitted,

Steven R. Gilliam, Reg. No. 51,734
Attorney for Applicant(s)
(512) 338-6320
(512) 338-6301 (fax)